8090B IEEE Transactions on Circuits and Systems
II: Analog and Digital Signal Processing
41(1994,February No.2, New York, US

Practical Considerations in the Implementation of a Frequency-Domain Adaptive Noise Canceller

Michael E. Deisher and Andreas S. Spanias

p. 164-168 (5)

Abstract— This paper is concerned with problems evolving around the implementation of a frequency-domain adaptive noise canceller on a fixed-point signal processor. In particular, we propose methods to improve the convergence speed and reduce the computational complexity of a constrained frequency-domain algorithm that uses a time-varying step size. In addition, we study the effects of finite word length and fixed-point arithmetic. Improvements are realized by adopting a new data reusing scheme and by applying running and pruned FFT's. Results are given using synthetic data, as well as data from noise cancellation experiments.

H03H 21/00 B5A

I. Introduction

The transversal LMS algorithm has been used extensively in many signal processing applications such as adaptive noise cancellation [1], [3] and active noise attenuation [2]. Two problems associated with the transversal LMS algorithm are poor convergence speed with correlated inputs and excessive computational requirements in high-order implementations. Frequency-domain adaptive filters [4], [5], [9] provide both for improved convergence and reduced computational complexity by using normalized convergence factors (μ's) and employing FFT's, respectively. A frequency-domain algorithm which adapts the filter coefficients using circular convolutions was proposed by Dentino *et al.* [4]. A constrained frequency-domain algorithm that implements the Block LMS algorithm [7] in the frequency domain (Fig. 1) was proposed in [5]. Considerable research effort has been devoted to the choice of μ [8], [9], [11] which controls the stability and the convergence speed of these algorithms. A time-varying μ for the Block LMS algorithm was proposed in [8]. A Fast Optimum Block Algorithm (FOBA) which also uses a time-varying μ was proposed in [9]. It was shown that the convergence factors proposed in [8], [9] resulted in improved convergence. Additionally, it was shown [8], [9] that considerable improvements in convergence speed are realized by processing blocks of data in an overlapping manner. These improvements, however, were realized at the expense of additional processing.

In this paper we propose: 1) methods to reduce the computational complexity of the FOBA, and 2) a new data-reusing scheme that enhances its convergence speed. In addition, we give results characterizing the effects of fixed-point arithmetic on the performance of the FOBA using synthetic, as well as real (noise cancellation) data. The paper is organized as follows. Section II describes the FOBA algorithm, while Section III discusses its convergence and computational complexity. Section IV discusses the effects of finite precision arithmetic, and Section V gives results from adaptive noise cancellation experiments. Finally, Section VI gives our conclusions.

II. The FOBA Algorithm

In this section, we describe the FOBA algorithm. The following tion is used. Lower and upper case symbols are used for time- frequency-domain variables, respectively. Matrix transposes and Hermitians (complex conjugate transposes) are denoted by ^{T} and ^{H}, respectively. The symbol $\langle\cdot,\cdot\rangle$ denotes the inner product, and diagonal matrices are denoted with a tilde.

The FOBA is different from the FLMS algorithm [5] in that it uses a time-varying step size. The taps of a frequency-domain FIR filter are adjusted to minimize the mean of the magnitude squared error. The N-point filter output y_k is obtained using the overlap-and-save technique [6], i.e.,

$$y_k = T_L \tilde{X}_k T_F A_k \tag{1}$$

where A_k is the $2N$-point complex filter weight vector, X_k is the DFT of the $2N$-point vector containing the present and previous block of input data, and

$$T_F = F\begin{bmatrix} I_{N\times N} & 0 \\ 0 & 0 \end{bmatrix}F^{-1} \qquad T_L = F\begin{bmatrix} 0 & 0 \\ 0 & I_{N\times N} \end{bmatrix}F^{-1}. \tag{2}$$

The $2N \times 2N$ DFT matrix is given by F, and 0 is an $N \times N$ zero matrix. The constrained gradient is obtained by taking the inverse FFT, setting the last N elements to zero, and taking the forward FFT, i.e.,

$$g_k = [\text{first } N \text{ terms of IFFT}\{-2\tilde{X}_k^H E_r\}] \tag{3}$$

$$G_k = \text{FFT}[g_k^T \mid 0^T]^T \tag{4}$$

where 0 is the N-point null vector. A compact form of the gradient is given by

$$G_k = -2T_F \tilde{X}_k^H E_k \tag{5a}$$

where

$$E_k = T_L(D_k - \tilde{X}_k T_F A_k) \tag{5b}$$

is the $2N$-point frequency-domain filter error vector and D_k is the DFT of the $2N$-point vector containing the present and previous block of the desired signal. The coefficient update is based on the complex LMS algorithm [12], i.e.,

$$A_{k+1} = A_k - \frac{\mu_k}{N} G_k. \tag{6}$$

μ_k is the adaptive step size for the kth block. An optimal adaptive step size for the constrained FLMS was given in [8], [9] by

$$\mu_k^0 = \frac{N\langle g_k, g_k\rangle}{2\langle c_k, c_k\rangle} \tag{7}$$

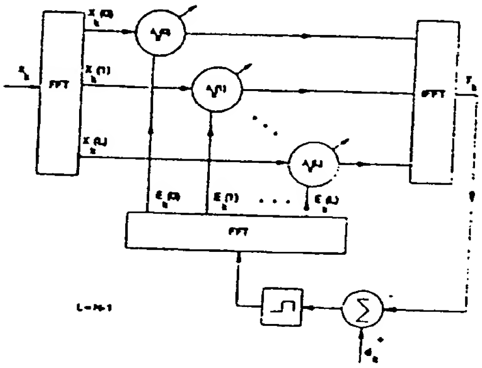

Fig. 1. Block diagram of the FLMS algorithm.

Manuscript received May 15, 1992; revised April 27, 1993. This work was supported in part by Intel Corporation Award 91PJM07. This paper was recommended by Associate Editor G. S. Moschytz.

The authors are with the Department of Electrical Engineering, Telecommunications Research Center, Arizona State University, Tempe, AZ 85287.

IEEE Log Number 9211977.

where

$$c_k = [\text{last } N \text{ terms of IFFT}\{\tilde{X}_k G_k\}] \tag{8}$$

or

$$c_k = F^{-1} T_L \tilde{X}_k G_k. \tag{9}$$

The adaptive step size is optimal in the sense that it minimizes an estimate of the block mean-square error from one iteration to the next. The derivation of the step size is given in [8]. Equations (3)–(9) describe the FOBA.

III. Convergence and Computational Complexity

In this section, we propose methods for improving the convergence speed and reducing the computational complexity of the FOBA. These involve replacing an FFT with the LMS spectrum analyzer [10], using packing and pruning FFT algorithms, and adopting simplified matrix-vector products. The $2N$-point vector x_k, which contains the present and previous blocks of input data, is updated m samples at a time. It can be shown [9] that if $m < N$, i.e., blocks are overlapped, the convergence speed of the FOBA improves. Block overlapping implies data reusing, and when $m = 1$, the convergence speed of the FOBA improves considerably [9]. Data reusing occurs because some of the samples in consecutive blocks are the same, and hence they are reused to adapt the coefficients in consecutive iterations. The idea of data reusing is exploited further here by performing r iterations *between* each new input sample when $m = 1$. For large r, significant improvement in convergence rate is realized at the expense of increased computation. The merit of this data-reusing scheme is demonstrated via a system identification simulation with $N = 32$ (see Fig. 2). In this simulation, the input x_k is white Gaussian noise (zero mean and unit variance) which drives an Nth-order fixed filter $F(z)$. The normalized error energy (NEE) is used to evaluate this simulation. This is given by

$$\text{NEE} = \frac{\int_{\omega=0}^{\omega=\pi} |F(e^{j\omega}) - A(e^{j\omega})|^2 \, d\omega}{\int_{\omega=0}^{\omega=\pi} |F(e^{j\omega})|^2 \, d\omega}. \tag{10}$$

Results are shown for (a) $m = N$, $r = 1$, (b) $m = 1$, $r = 1$, (c) $m = 1$, $r = 10$, and (d) $m = 1$, $r = 100$. Fig. 2 shows that as r increases, the convergence speed improves. For large r, convergence time was generally found to approach N samples. Note that the improvement observed here is in terms of convergence time, and not misadjustment. In fact, it is well known [1] that the convergence speed of LMS-type algorithms can be improved only at the expense of increased misadjustment. Hence, the result of Fig. 2 is consistent with the theory of convergence of LMS algorithms.

Methods to reduce the computational complexity of the FOBA are now presented. Diagonal matrix–vector multiplications require only $N + 1$ products since the frequency-domain data are complex conjugate symmetric. When $m = 1$, the FFT of x_k may be replaced by the adaptive "LMS spectrum analyzer" ($\mu = 0.5$) described by Widrow *et al.* [10]. This algorithm provides an "adaptive" running DFT of the input for each new sample. The algorithm is described below, i.e.,

$$V_k = [1 \quad e^{j(2\pi k/2N)} \cdots e^{j2\pi k(2N-1)/2N}]^T \tag{11}$$

$$W_k = \sum_{m=k-2N}^{k-1} x_m V_m^*, \qquad (m \geq 0) \tag{12}$$

$$X_{k-1} = \tilde{V}_k W_k \tag{13}$$

where V_k is a vector of harmonic phasors, W_k is a vector of complex adaptive weights, X_k is the transform of the last $2N$ samples of the input x_k, and j is $\sqrt{-1}$. These results can be simplified further by expressing the complex weight update equation (12) as

$$\begin{aligned} \mathbf{W}_k = {} & \mathbf{W}_{k-1} - x_{k-2N-1} \\ & \cdot \mathbf{V}^*_{k-2N-1} + x_{k-1} \mathbf{V}^*_{k-1}, \qquad k \geq 2N. \end{aligned} \tag{14}$$

Since V_k is periodic with period $2N$, $V_k = V_{k-2N}$ and (14) becomes

$$W_{k+1} = W_k + (x_k - x_{k-2N}) V_k^*, \qquad k \geq 2N. \tag{15}$$

Substituting (15) into (13) and simplifying yields

$$X_k = \tilde{V}_1 X_{k-1} + \tilde{V}_1 (x_k - x_{k-2N}). \tag{16}$$

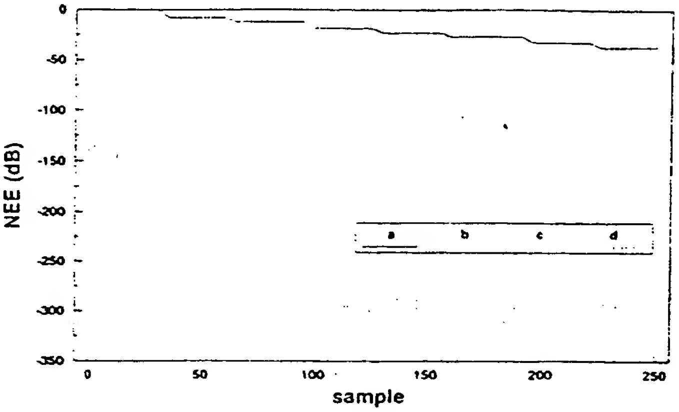

Fig. 2. Comparison of results from FOBA simulation. The curves correspond to the following: (a) $m = N$, $r = 1$, (b) $m = 1$, $r = 1$, (c) $m = 1$, $r = 10$, and (d) $m = 1$, $r = 100$.

Thus, when $m = 1$, the FFT of x_k may be replaced by the modified "steady flow DFT" (SFDFT) given by (16) which requires only $2N$ complex multiplications and $2N + 1$ complex additions. If x_k is real, only $N + 1$ complex multiplications and $N + 2$ complex additions are required. V_1 may be recognized as a twiddle factor vector required by FFT algorithms. Therefore, this technique introduces no additional storage requirements.

The bulk of the remaining computations required by the FOBA are contained in five other FFT's. The SFDFT cannot be applied for these transformations since all the time-domain samples are updated on a block-by-block basis. They may be simplified, however, by eliminating unnecessary computations. The overlap-and-save technique requires that half the inputs to the FFT's used to compute E_k and G_k be zero. From results presented by Skinner [13], it may be shown that the number of computations required to perform these FFT's is reduced by 50% when the DIT FFT algorithm is "pruned" to avoid performing butterflies with zero inputs. The overlap-and-save technique also requires that half the output points from the FFT's used to compute y_k, g_k, and c_k be ignored. Since butterfly diagrams of the DIF and DIT FFT algorithms are essentially mirror images of each other (see Fig. 3), Skinner's results may be applied to prune the output of the DIF algorithm to avoid computing the unused output points [14]. If the samples x_k are real valued, then all FFT's operate on either real or complex conjugate symmetric vectors. It has been shown [15] that an N-point FFT may be used to compute the transform of a $2N$-point real sequence. $O(N)$ computations are then required to "unpack" the $2N$-point complex result. The pruned FFT

TABLE I
COMPUTATIONS/SAMPLE ($m = N$, $r = 1$)

Algorithm Modification	Real Multiplications[a] Per Sample	Real Additions[a] Per Sample
Complex FOBA	$24 \text{ LOG}_2 N + 54$	$36 \text{ LOG}_2 N + 54$
Complex FOBA (PFFT's)	$14 \text{ LOG}_2 N + 44$	$21 \text{ LOG}_2 N + 39$
Real FOBA	$12 \text{ LOG}_2 N + 39 + 38/N$	$18 \text{ LOG}_2 N + 57 + 55/N$
Real FOBA (PFFT's)	$7 \text{ LOG}_2 N + 29 + 28/N$	$10.5 \text{ LOG}_2 N + 37 + 35/N$

[a] Considering that one complex multiply = four real multiplications and two real adds.

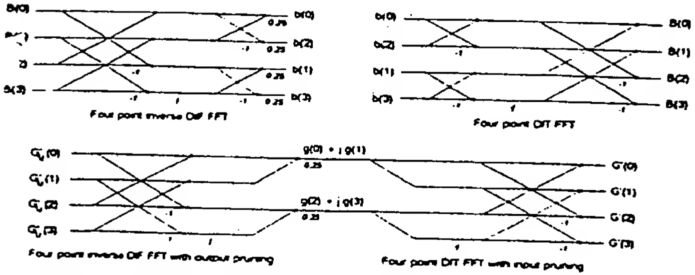

Fig. 3. Gradient constraint via packing and pruning techniques.

(PFFT) algorithms can be used for the N-point FFT required above for approximately 75% reduction in computation. As an example, Fig. 3 illustrates how the packing and pruning technique is used to constrain the gradient when $N = 4$.

This is the windowing process shown in (3) and (4). The eight elements of the unconstrained frequency-domain gradient G_u are "packed" (yielding G'_u) into the input bins of the four-point FFT. The inverse FFT is performed, and unnecessary computations are eliminated. Odd elements of the time-domain gradient vector appear as the complex part, and even elements as the real part. The forward FFT is then performed yielding G', which is the "packed" version of the constrained frequency-domain gradient. Table I illustrates the computational complexity of the FOBA with ordinary and pruned FFT's (PFFT's). The complexity is described in terms of the additions and multiplications required per input sample. Both real or complex-valued input data are considered. As shown in Table I, the use of PFFT's reduces the complexity of the FOBA significantly.

When $m = 1$, we have shown that the convergence speed may be significantly improved at the expense of increased computational complexity. This becomes evident when the entries of Table I are compared against those of Table II. Notice that in both tables, the complexity is expressed in terms of computations per input sample as opposed to input block. This allows direct comparison of the FOBA and the data-reusing FOBA. Table II shows the effect of the two convergence improvements discussed (i.e., block overlapping and r extra iterations per sample) on the computational requirements of the FOBA. Table II also illustrates the computational savings obtained when $m = 1$ and the SFDFT is used in place of an FFT.

IV. EFFECTS OF FIXED-POINT ARITHMETIC

This section examines some of the problems associated with the mplementation of the FOBA on a fixed-point signal processor. The .MS algorithm can become numerically unstable, even when finite recision floating-point schemes are used. Fixed-point arithmetic omplicates this problem because it is more susceptible to roundoff, verflow, and underflow error than floating-point arithmetic. The et of fixed-point machine numbers is more dense than the set of oating-point numbers (assuming the same word length). However, ie maximum representable magnitude is generally much smaller, and the minimum representable magnitude is generally much larger in fixed-point than in floating-point arithmetic. Although values within this range are represented more accurately in fixed-point arithmetic, operations whose result has a different order of magnitude than its operands can result in overflow, underflow, or roundoff. Fig. 4 shows the effect of fixed-point arithmetic on the adaptation of the FOBA. These results are from a system identification simulation with $N = 32$, $m = 1$, and $r = 1$ using 16-, 24-, and 32-bit fixed-point and 32-bit floating-point arithmetic. For the 24-bit case, the Motorola DSP56000 Mixed Number format [16] was used, i.e., an 8-bit integer and a 16-bit fraction.

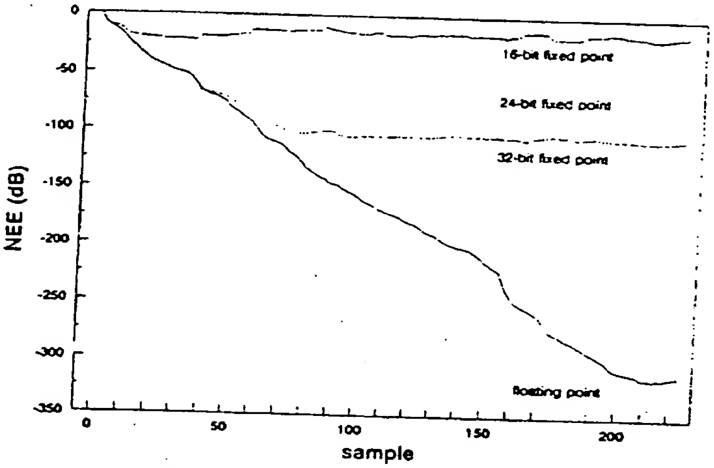

Fig. 4. Effects of fixed-point arithmetic on FOBA adaptation.

Several adjustments were made to prevent fixed-point overflow and underflow. The input sequences d_k and x_k were normalized to prevent overflow. The FOBA in this case was modified to prevent some of the adverse effects of fixed-point arithmetic. The normalized FFT was used to control the magnitude of vectors in the frequency domain. All the FFT's were replaced by normalized FFT's, except for the one used to compute X_k. The computation of μ_k shown in (7) involves dividing one inner product by another. The inner product in the denominator can be expressed as a matrix product with four terms, two of which depend on the filter error. When the filter error is small, underflow can cause this result to become zero. When the filter error is large, the inner products may result in fixed-point overflow. To avoid overflow and underflow, each term of each inner product summation was scaled by $\max_i c_k^2(i)$, $N \leq i < 2N$. If either inner product resulted in underflow, μ_k was not updated. Clearly, additional computational overhead ($2N$ additional multiplies) and roundoff error was introduced in order to safely calculate μ_k. For high-order filters, this is not a significant increase. When processor speed limitations require that lower order filters be used, a fixed μ_k may be necessary.

TABLE II
COMPUTATIONS/SAMPLE ($m = 1$)

Algorithm Modification	Real Multiplications[a] Per Sample	Real Additions[a] Per Sample
Complex FOBA ($r = 1$)	$24N \log_2 N + 54N$	$36N \log_2 N + 54N$
Real FOBA ($r = 1$)	$12N \log_2 N + 39N + 38$	$18N \log_2 N + 57N + 55$
Real FOBA (SFDFT)	$10rN \log_2 N + (35r + 0.5)N + 34r + 1$	$15rN \log_2 N + (49r + 0.5)N + 46r + 2$
Real FOBA (SFDFT, PFFT's)	$5rN \log_2 N + (35r + 0.5)N + 34r + 1$	$7.5rN \log_2 N + (49r + 0.5)N + 46r + 2$

[a] Considering that one complex multiply = four real multiplications and two real adds.

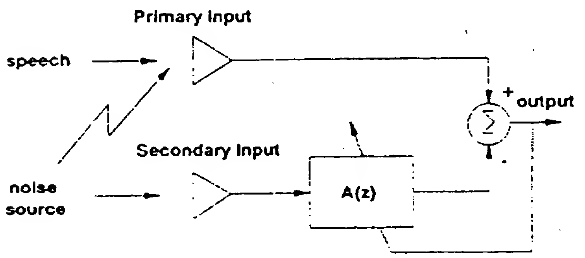

Fig. 5. Adaptive noise canceller.

V. THE FOBA FOR ADAPTIVE NOISE CANCELLATION

The FOBA is used here for adaptive noise cancellation (Fig. 5) in conjunction with the SFDFT (when $m = 1$). The weights are adjusted to minimize the difference between the primary input and the adaptive filter output. When the speech is uncorrelated with the noise, this difference is a "best" least-squares estimate of the speech signal [1].

Experimental results were obtained using a white noise generator in an anechoic chamber. The microphones were one foot apart. The two inputs were sampled at 8 kHz and quantized to 11 b. Filtering was done off-line using both floating-point and fixed-point arithmetic. The phrase, "Good evening, I'm Ted Koppel and this is Nightline. Tonight...," was spoken into the primary microphone. Fig. 6 shows the segmental energy (SE) of the noise canceller output and that of primary input. This is obtained from

$$\mathrm{SE}(x, k) = \frac{1}{N} \sum_{n=k}^{k+N-1} x^2(n) \tag{17}$$

where x is the signal being evaluated. The original (noisy) signal is shown as segmental energy (a). The FOBA ($N = 128$, $m = 1$, $r = 1$) with floating-point arithmetic was used to adjust the filter weights resulting in segmental energy (d). The noise energy [compare (a) and (d)] is shown to be attenuated by approximately 25 dB. Segmental energies (b) and (c) were generated using the 24-bit DSP 56000 Mixed Number fixed-point format. For the data-reusing FOBA, background noise grew noticeably louder as time elapsed [see segmental energy (b)]. This illustrates the effect of roundoff error accumulation when the data-reusing schemes discussed earlier are implemented on a fixed-point signal processor. When disjoint blocks were used ($m = 128$), segmental energy (c) was obtained, and this effect was not as evident. Approximately 10 dB noise reduction was achieved in the latter case.

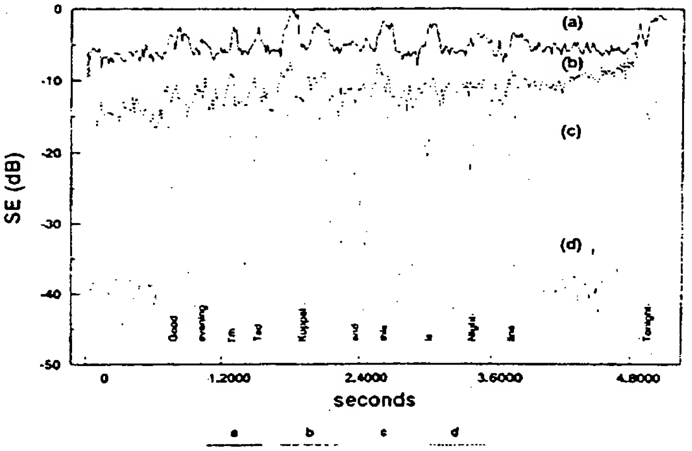

Fig. 6. Comparison of segmental signal energies. The curves correspond to the following: (a) segmental energy for noisy speech, (b) segmental energy of the output of a fixed-point ANC with overlapping blocks ($N = 128$, $m = 1$, $r = 1$), (c) segmental energy of the output of a fixed-point ANC with disjoint blocks ($N = 128$, $m = 128$, $r = 1$), and (d) segmental energy of the output of a floating-point ANC with overlapping blocks ($N = 128$, $m = 1$, $r = 1$).

VI. CONCLUDING REMARKS

In this paper, we studied implementation issues associated with the fixed-point realization of a frequency-domain adaptive algorithm. In particular, we suggested methods for improving the convergence speed and reducing the computational complexity of the Frequency-Domain Optimum Block Algorithm (FOBA). Improvements in the convergence speed (in terms of data samples) were realized by performing r weight updates using the same block of data. Simulations using floating-point arithmetic, white noise inputs, and large r have generally shown that the number of samples required for the FOBA to identify an FIR filter of the same order was roughly equal to the length of its impulse response. The computational complexity of the FOBA was reduced by using the SFDFT and packing and pruning FFT's.

We also examined the proposed methods in a more realistic scenario by implementing a frequency-domain adaptive noise canceller that uses the FOBA. Results were given for fixed- and floating-point arithmetic. Although convergence improvements, using the aforementioned methods, were evident using floating-point arithmetic, fixed-point implementation introduced fixed-point roundoff error accumulation which had an adverse effect on the performance of the noise canceller. Experiments also revealed that the computation of μ_k is sensitive to fixed-point implementation.

REFERENCES

[1] B. Widrow *et al.*, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, p. 1692, Dec. 1975.

[2] J. C. Burgess, "Active adaptive sound control in a duct: A computer simulation," *J. Acoust. Soc. Amer.*, vol. 70, p. 715, Sept. 1981.

[3] W. B. Mikhael and P. Hill, "Performance evaluation of a real time adaptive noise canceller," *IEEE Trans. Acoust., Speech, Signal Proc.*, Mar. 1988.

[4] M. Dentino *et al.*, "Adaptive filtering in the frequency domain," *Proc. IEEE*, vol. 66, p. 1658, Dec. 1978.
[5] E. R. Ferrara, "Fast implementation of LMS adaptive filters," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-28, p. 474, Aug. 1980.
[6] A V Oppenheim and R. W. Schafer, *Digital Signal Proc.*. Englewood Cliffs, NJ, Prentice-Hall, 1975.
[7] G A Clark *et al.*, "Block implementation of LMS adaptive filters," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-29, p. 744, June 1981
[8] W B Mikhael and F H. Wu, "Fast algorithms for block FIR adaptive digital filtering," *IEEE Trans. Circuits Syst.*, vol. CAS-34, p. 1152, Oct. 1987
[9] W B Mikhael and A. S Spanias, "A fast frequency-domain adaptive algorithm," *Proc IEEE*, vol. 76, p. 80, Jan. 1988.
[10] B Widrow *et al* "Fundamental relations between the LMS algorithm and the DFT," *IEEE Trans. Circuits Syst.*, vol. CAS-34, p. 814, July 1987
[11] W B Mikhael and F F. Yassa, "Optimality in the choice of the convergence factors for adaptive algorithms," in *Proc. IEEE Conf. ISCAS 83* vol 3 Newport Beach, CA, May 1983, p. 1367.
[12] B Widrow *et al* "The complex LMS algorithm," *Proc. IEEE*, vol. 63, p. 719, Apr 1975
[13] D P Skinner "Pruning the decimation-in-time FFT algorithm," *IEEE Trans Acoust Speech Signal Proc*, vol. ASSP-24, p. 193, Apr. 1976.
[14] T. V Sreenivas and P V S Rao, "FFT algorithm for both input and output pruning" *IEEE Trans Acoust., Speech, Signal Proc.*, vol. ASSP-27 p 291 June 1979
[15] J. W Cooley *et al* "The FFT algorithm: Programming considerations in the calculation of sine cosine and Laplace transforms," *J. Sound Vib.*, vol. 12 no 3, p 315, 1970
[16] A. Chrysafis and S Lansdowne, *Fractional and integer arithmetic using the DSP56000 family of general purpose digital signal processors*, Motorola Inc 1988

Quantization Noise Spectrum of Double-Loop Sigma-Delta Converter with Sinusoidal Input

Sundeep Rangan and Bosco Leung

Abstract—**An exact formula for the output noise spectrum of a double-loop sigma–delta modulator, under the no overloading assumption and with a sinusoidal input, is derived without the use of a white-noise model. In the case of a sinusoidal input with irrational input amplitude and digital frequency, the result agrees with the exact formula derived by ergodic theory for two-stage modulators. In addition, the present method also provides an exact formula for a sinusoidal inputs with rational frequency and amplitude. Furthermore, the period of the output with rational initial conditions and dc input is also calculated. The results are of primary interest to multibit sigma–delta modulators, which do not overload over the entire input amplitude range.**

The ergodic theory method for calculating the exact noise spectrum involves explicitly determining the autocorrelation of the internal quantization error with ergodic theory techniques, and then determining the noise spectrum from the correlation function. The present method, however, directly determines the quantization noise spectrum by using an open-loop model for the coder and applying a Fourier series representation of the quantization error function.

The result of both of these methods is that the output noise spectrum f[illegible]inusoidal input is composed of discrete spectral lines shaped by a si[illegible]/2) envelope.

Manuscript received April 22, 1992; revised May 21, 1993. This paper was recommended by Associate Editor B. S. Song.

The authors are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ont., Canada N2L 3G1.

IEEE Log Number 9211976

I. Introduction

The quantization noise spectrum is an important characteristic of analog-to-digital (A/D) and digital-to-analog (D/A) converters. However, for oversampled converters, such as sigma–delta modulators, the exact determination of this spectrum is made difficult by the presence of a nonlinear quantizer in the feedback loop. To overcome this difficulty, most previous analyses are based on replacing the quantizer with an additive white noise source which is assumed to be uncorrelated with the input and uniformly distributed. This results in a linear model for the system, which can be easily analyzed by standard linear system techniques [1], [2]. This paper presents, without the use of the white-noise model and under the no overloading assumption, an exact formula for the quantization noise spectrum for a double-loop sigma–delta coder with a sinusoidal input.

The previous exact analysis of the double-loop output quantization noise spectrum for dc irrational inputs, assuming no overloading, was done in [8], where it is demonstrated that the output quantization noise is as predicted by the white-noise model. A similar exact analysis for the sinusoidal input case for two-stage sigma–delta coders was done in [11] where it was shown that the quantization noise spectrum consists of discrete spectral lines. Two-stage and a nonoverloading double-loop coder were shown to be mathematically equivalent in [7]. A 2-b quantizer is sufficient to guarantee no overloading over the entire input amplitude range [7], [8]. Practical implementations of multibit sigma–delta modulators have received significant attention recently because a higher SNR (signal-to-noise ratio) as well as lower out-of-band noise can be obtained [12]–[15].

The previous exact analyses are based on applying ergodic theory to explicitly determine the asymptotic autocorrelation of the quantization error sequence, and then to find the quantization noise spectrum from the autocorrelation function. A difficulty with this method is that assumptions must be made with respect to the irrationality of some of the input parameters. However, it is shown in [3]–[5] that rational inputs may give rise to limit cycles of the coder, and the output noise tends to peak near some of these input rational values. Consequently, the rational inputs may be relevant to understanding certain quantization noise phenomena. This is particularly the case for D/A converters where the input is rational.

In this paper, we present an alternative, direct derivation of the exact double-loop sigma–delta quantization noise with sinusoidal input assuming no overloading. The method used here is to apply a Fourier series representation of the quantization error function, employed previously in the analysis of single-loop delta modulators [9]. In the case of a sinusoidal input with irrational frequency and amplitude, the result agrees with the ergodic theory results. However, an exact formula for the output noise spectrum is also provided for inputs with rational frequency and amplitude. Furthermore, the period of the output is explicitly calculated for a coder with rational initial conditions and rational dc input.

In section II, we present the architecture of the sigma–delta modulator under study. In section III, an open-loop model, similar to the one used in the ergodic theory analyses, is developed for the sigma–delta coder. In section IV, the noise spectrum formula is derived based on replacing the quantizer in the open-loop model by its Fourier series representation. In section V, this noise formula is compared to the formula predicted by the white noise and the ergodic theory result. Simulation results, which confirm the analysis, are also presented in this section.

1057-7130/94$04.00 © 1994 IEEE